



MODUL PRAKTIKUM

PEMROGRAMAN BERBASIS FRAMEWORK

Modul 5

- Cara menginstall react router – Basic Router
- URL Parameters
- Use Nesting Router
- Use Redirects (Auth)

POLITEKNIK NEGERI MALANG

React Router

Dalam web jika ingin berganti halaman satu dan halaman yang lainnya diperlukan suatu proses routing. Routing sendiri adalah proses pemetaan antara sebuah URL ke dalam sebuah halaman yang terdapat konten / UI (User Interface) sesuai dengan URL yang dituju. Jika ingin membuat routing membutuhkan library tambahan karena tidak secara langsung tersedia. Ternyata ada beberapa library yang bisa digunakan, diantara library yang sangat familiar adalah *react-router* dan *reach/router*. Sebenarnya yang dipakai untuk routing di React biasanya *react-router-dom* anak dari *react-router*, yang mana selain *react-router-dom* juga terdapat *react-router-native* yang bisa digunakan untuk development aplikasi Android dan iOS. Library *react-router-dom* dapat diinstall dengan menjalankan perintah

```
yarn add react-router-dom
```

atau

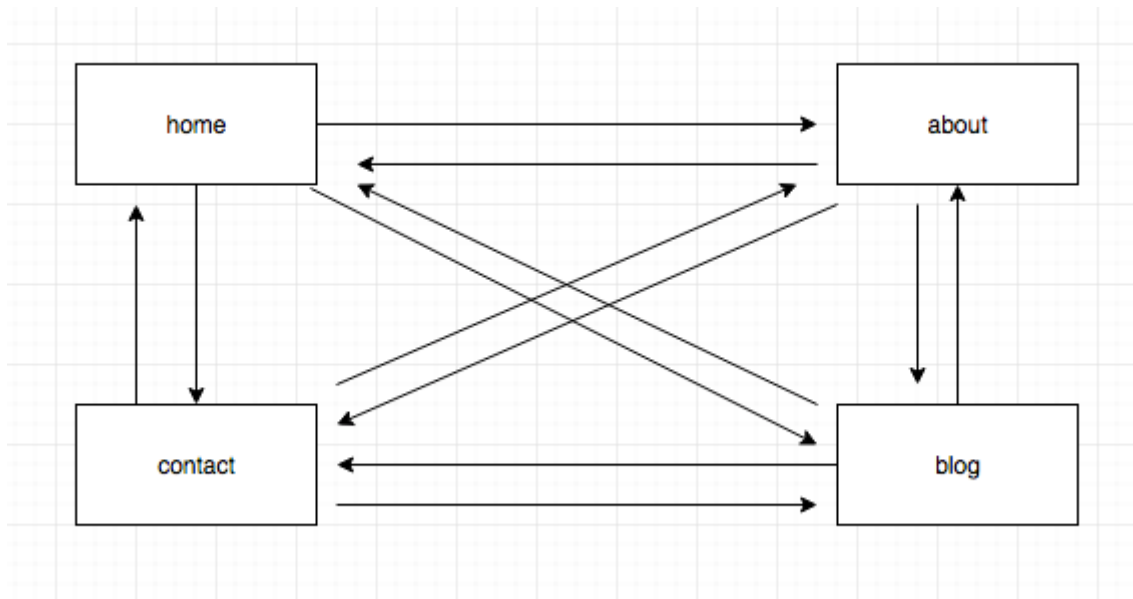
```
npm i react-router-dom
```

Pada dasarnya, *react-router-dom* mempunyai 2 jenis router yang dapat kita gunakan, yaitu *HashRouter* dan *BrowserRouter*. Keduanya mempunyai kegunaan masing-masing tergantung dari jenis Web apa yang akan kita buat. Seperti contoh, jika kita ingin membuat sebuah web yang static atau tidak ada server untuk me-render data yang dinamis, maka sebaiknya kita menggunakan *HashRouter*. Sebaliknya, jika kita membuat web yang menggunakan data dinamis dengan server backend, maka menggunakan *BrowserRouter* adalah pilihan tepat.

Sebagai contoh dalam membangun aplikasi katalog sederhana dimana user dapat berpindah – pindah secara dinamis dari satu halaman ke halaman lain. Daftar halaman yang biasanya dapat diakses oleh user adalah:

- Home atau landingpage
- About, halaman untuk menampilkan informasi mengenai website tersebut.
- Contact, halaman yang menampilkan kontak(pemiliki atau pengelola website) yang dapat dihubungi
- Blog, halaman untuk menampilkan posting atau berita.

Navigasi akan diletakan pada komponen menu sehingga perpindahan dapat berasal dari halaman mana saja ke tujuan mana saja. Berikut ini adalah ilustrasi skema navigasi diatas.



Gambar 1. Skema Navigasi

dari ilustrasi diatas kita akan membuat lima buah komponen yakni :

- komponen home
- komponen about
- komponen blog
- komponen kontak
- komponen menu

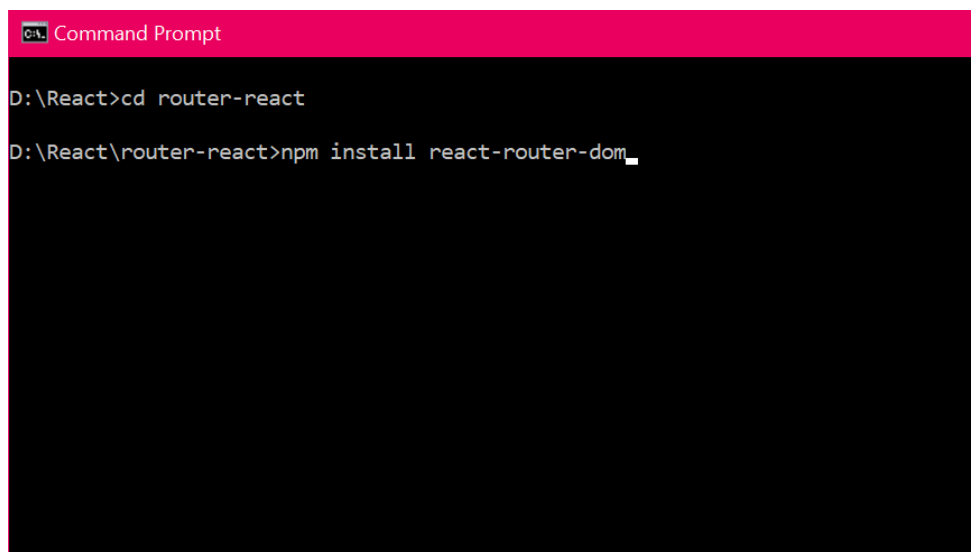
Praktikum

Dalam membangun web SPA (*Single Page Application*) kita membutuhkan library React Router. Untuk itu langkah pertama yang kita lakukan adalah menginstall library React Router.

1.1 Langkah Praktikum

a. Basic Router

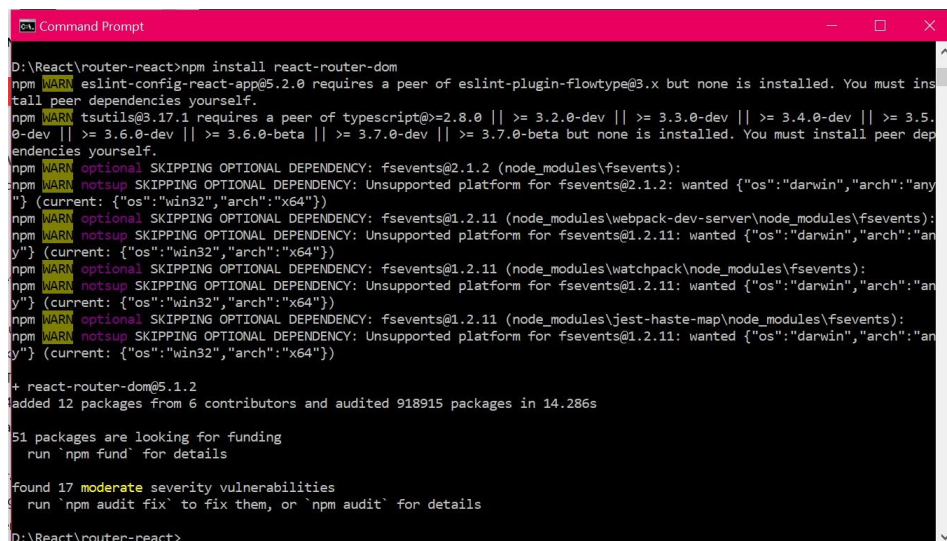
1. Setelah kita sukses melakukan instalasi react (*npm install*), kita perlu menambahkan library react router dengan menambahkan *npm install react-router-dom*.



```
Command Prompt

D:\React>cd router-react

D:\React\router-react>npm install react-router-dom
```



```
Command Prompt

D:\React\router-react>npm install react-router-dom
npm WARN eslint-config-react-app@5.2.0 requires a peer of eslint-plugin-flowtype@3.x but none is installed. You must ins
tall peer dependencies yourself.
npm WARN tsutils@3.17.1 requires a peer of typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev || >= 3.5
.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta but none is installed. You must install peer dep
endencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.1.2 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.1.2: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.11 (node_modules\webpack-dev-server\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.11: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.11 (node_modules\watchpack\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.11: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.11 (node_modules\jest-haste-map\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.11: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})
+ react-router-dom@5.1.2
added 12 packages from 6 contributors and audited 918915 packages in 14.286s

51 packages are looking for funding
  run `npm fund` for details

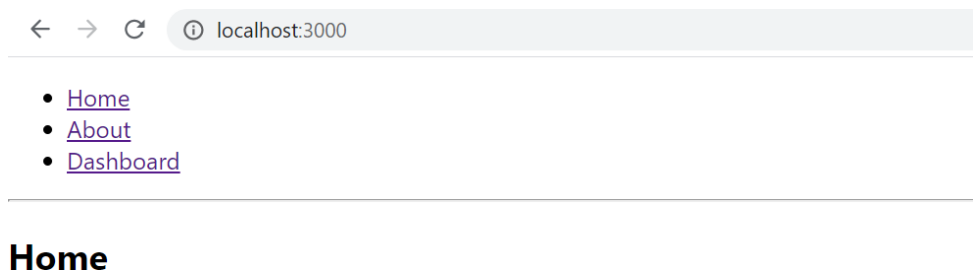
found 17 moderate severity vulnerabilities
  run `npm audit fix` to fix them, or `npm audit` for details

D:\React\router-react>
```

2. Kemudian buka app.js pada dan ketikkan code dibawah ini.

```
1  import React from "react";
2  import {
3    BrowserRouter as Router,
4    Switch,
5    Route,
6    Link
7  } from "react-router-dom";
8
9  // Situs ini memiliki 3 halaman, yang semuanya dirender secara dinamis di browser.
10 // Meskipun halaman tidak pernah di-refresh, perhatikan bagaimana React Router
11 // membuat URL selalu terbaru saat Anda menavigasi situs.
12 // Ini menjaga riwayat browser, memastikan hal-hal seperti tombol kembali
13 // dan bookmark berfungsi dengan baik.
14
15 export default function BasicExample() {
16   return (
17     <Router>
18       <div>
19         <ul>
20           <li>
21             <Link to="/">Home</Link>
22           </li>
23           <li>
24             <Link to="/about">About</Link>
25           </li>
26           <li>
27             <Link to="/dashboard">Dashboard</Link>
28           </li>
29         </ul>
30         <hr />
31
32         <Switch>
33           <Route exact path="/">
34             <Home />
35           </Route>
36           <Route path="/about">
37             <About />
38           </Route>
39           <Route path="/dashboard">
40             <Dashboard />
41           </Route>
42         </Switch>
43       </div>
44     </Router>
45   );
46 }
47
48 // Anda dapat menganggap komponen ini sebagai "halaman(konten)" di situs Anda.
49
50 function Home() {
51   return (
52     <div>
53       <h2>Home</h2>
54     </div>
55   );
56 }
57
58 function About() {
59   return (
60     <div>
61       <h2>About</h2>
62     </div>
63   );
64 }
65
66 function Dashboard() {
67   return (
68     <div>
69       <h2>Dashboard</h2>
70     </div>
71   );
72 }
```

3. Lakukan npm start, lalu cek hasilnya akan seperti dibawah ini.



Pilih setiap menu lalu amati bagaimana perubahannya.

b. URL Parameters

1. Pada percobaan kali ini kita mencoba membuat link react router dengan menggunakan params, sehingga kita hanya membuat satu template yang dapat berubah kontennya sesuai dengan apa yang kita klik.
2. Ketikkan seperti code dibawah ini.

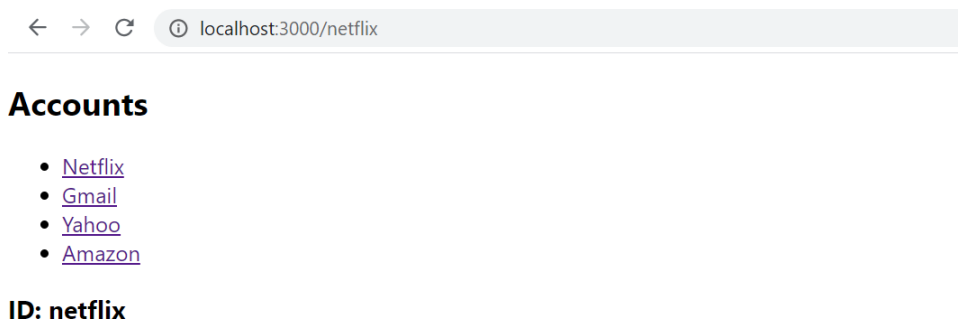
```
1  import React from "react";
2  import {
3    BrowserRouter as Router,
4    Switch,
5    Route,
6    Link,
7    useParams
8  } from "react-router-dom";
9
10 // Params adalah placeholder di URL yang dimulai dengan titik dua,
11 //seperti param `: id` yang didefinisikan dalam route dalam contoh ini.
12 export default function ParamsExample() {
13   return (
14     <Router>
15       <div>
16         <h2>Accounts</h2>
17         <ul>
18           <li>
19             <Link to="/netflix">Netflix</Link>
20           </li>
21           <li>
22             <Link to="/gmail">Gmail</Link>
23           </li>
24           <li>
25             <Link to="/yahoo">Yahoo</Link>
26           </li>
27           <li>
28             <Link to="/amazon">Amazon</Link>
29           </li>
30         </ul>
31       </div>
32     </Router>
33   );
34 }
```

```

31
32     <Switch>
33       <Route path="/:id" children={<Child />} />
34     </Switch>
35   </div>
36 </Router>
37 );
38 }
39
40 function Child() {
41   let { id } = useParams();
42
43   return (
44     <div>
45       <h3>ID: {id}</h3>
46     </div>
47   );
48 }

```

3. Cek hasilnya maka akan seperti dibawah ini.



Secara tampilan memang hasilnya hampir sama dengan praktikum sebelumnya, namun dengan metode yang berbeda.

c. Use Nesting Router

1. Jika kita ingin melakukan percabangan link di dalam react router, maka kita dapat menggunakan cara ini sebagai solusinya.
2. Ketikkan seperti code dibawah ini.

```

1  import React from "react";
2  import {
3    BrowserRouter as Router,
4    Switch,
5    Route,
6    Link,
7    useParams,
8    useRouteMatch
9  } from "react-router-dom";
10
11 // Karena route adalah komponen regular react,
12 //sehingga dapat ditampilkan di mana saja dalam penempatannya,
13 //termasuk dalam child element.
14
15 // Ini akan membantu anda untuk memecah menjadi beberapa bundel
16 //karena pemisahan kode pada aplikasi React Router sama dengan
17 //pemecahan kode pada aplikasi react lainnya.
18
19 export default function NestingExample() {
20   return (
21     <Router>
22       <div>
23         <ul>
24           <li>
25             <Link to="/">Home</Link>
26           </li>
27           <li>
28             <Link to="/topics">Topics</Link>
29           </li>
30         </ul>
31
32         <hr />
33
34         <Switch>
35           <Route exact path="/">
36             <Home />
37           </Route>
38           <Route path="/topics">
39             <Topics />
40           </Route>
41         </Switch>
42       </div>
43     </Router>
44   );
45 }
46
47 function Home() {
48   return (
49     <div>
50       <h2>Home</h2>
51     </div>
52   );
53 }

```



```

55 function Topics() {
56   // `path` untuk membuat jalur <Route> yang terhadap rute induk,
57   //sedangkan `url` untuk membuat link.
58   let { path, url } = useRouteMatch();
59   return (
60     <div>
61       <h2>Topics</h2>
62       <ul>
63         <li>
64           <Link to={`${url}/Sate, Nasi goreng`}>Kuliner</Link>
65         </li>
66         <li>
67           <Link to={`${url}/Wisata alam, Museum`}>Travelling</Link>
68         </li>
69         <li>
70           <Link to={`${url}/Ibis, JW Marriot`}>Review Hotel</Link>
71         </li>
72       </ul>
73
74       <Switch>
75         <Route exact path={path}>
76           <h3>Please select a topic.</h3>
77         </Route>
78         <Route path={`${path}/:topicId`} >
79           <Topic />
80         </Route>
81       </Switch>
82     </div>
83   );
84 }

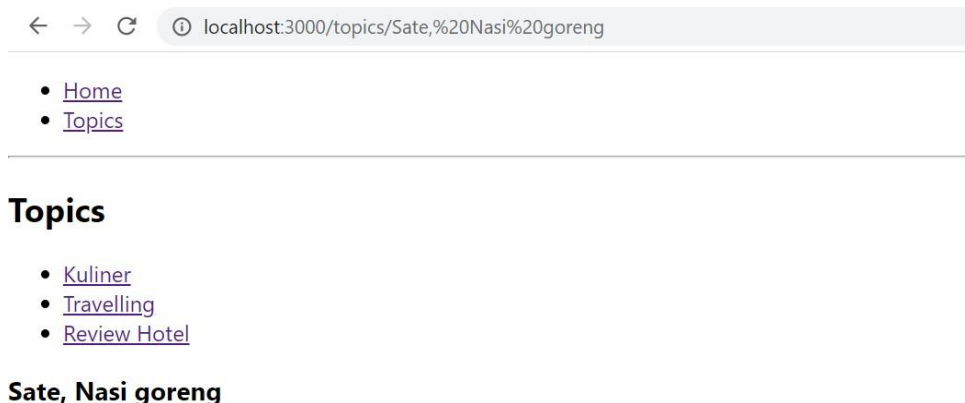
```

```

85
86 function Topic() {
87   let { topicId } = useParams();
88
89   return (
90     <div>
91       <h3>{topicId}</h3>
92     </div>
93   );
94 }

```

3. Cek hasilnya dan coba klik setiap link nya.



Bisa kita lihat bahwa link induk yang diatas akan menentukan hasil yang keluar dibawahnya, dan link yang dibawah dapat memunculkan hasil yg berbeda pula.

d. Use Redirects (Auth)

1. Cara ini dapat anda gunakan untuk masuk ke halaman yang membutuhkan authentication. Setelah masuk dan melakukan klik ke menu lain termasuk masuk ke halaman yang memerlukan authentication, posisi user tidak perlu login ulang (karena posisi sudah logged) sampai user melakukan sign out.
2. Ketikkan seperti code dibawah in.

```
1  import React from "react";
2  import {
3    BrowserRouter as Router,
4    Switch,
5    Route,
6    Link,
7    Redirect,
8    useHistory,
9    useLocation
10 } from "react-router-dom";
11
12 // Pada aplikasi ini memiliki 3 halaman: public page, private page, dan halaman login.
13 //Untuk masuk ke private page, Anda harus login terlebih dahulu.
14
15 // Pertama, klik public page. Kemudian, kunjungi private page.
16 //karena anda belum login, jadi Anda diarahkan ke halaman login.
17 //Setelah login, Anda akan diarahkan kembali ke private page.
18
19 // Perhatikan perubahan setiap URL. Jika Anda mengklik tombol kembali,
20 //apakah anda kembali ke halaman login? Tidak, karena anda sudah login.
21 //Cobalah, maka anda akan kembali ke halaman yang Anda kunjungi sebelum login, yaitu public page.
22
```

```
23 export default function AuthExample() {
24   return (
25     <Router>
26       <div>
27         <AuthButton />
28
29         <ul>
30           <li>
31             <Link to="/public">Public Page</Link>
32           </li>
33           <li>
34             <Link to="/private">Private Page</Link>
35           </li>
36         </ul>
37
38         <Switch>
39           <Route path="/public">
40             <PublicPage />
41           </Route>
42           <Route path="/login">
43             <LoginPage />
44           </Route>
45           <PrivateRoute path="/private">
46             <ProtectedPage />
47           </PrivateRoute>
48         </Switch>
49       </div>
50     </Router>
51   );
52 }
```

```

54 const fakeAuth = {
55   isAuthenticated: false,
56   authenticate(cb) {
57     fakeAuth.isAuthenticated = true;
58     setTimeout(cb, 100); // fake async
59   },
60   signout(cb) {
61     fakeAuth.isAuthenticated = false;
62     setTimeout(cb, 100);
63   }
64 };
65
66 function AuthButton() {
67   let history = useHistory();
68
69   return fakeAuth.isAuthenticated ? (
70     <p>
71       Welcome!{" "}
72       <button
73         onClick={() => {
74           fakeAuth.signout(() => history.push("/"));
75         }}
76       >
77         Sign out
78       </button>
79     </p>
80   ) : (
81     <p>You are not logged in.</p>
82   );
83 }

```

```

85 // Pembungkus untuk <Route> yang mengarahkan ke login
86 // tampilan jika Anda belum terkonfirmasi.
87
88 function PrivateRoute({ children, ...rest }) {
89   return (
90     <Route
91       {...rest}
92       render={({ location }) =>
93         fakeAuth.isAuthenticated ? (
94           children
95         ) : (
96           <Redirect
97             to={{
98               pathname: "/login",
99               state: { from: location }
100             }}
101           />
102         )
103       >
104     </Route>
105   );
106 }
107
108 function PublicPage() {
109   return <h3>Public</h3>;
110 }
111
112 function ProtectedPage() {
113   return <h3>Private</h3>;
114 }

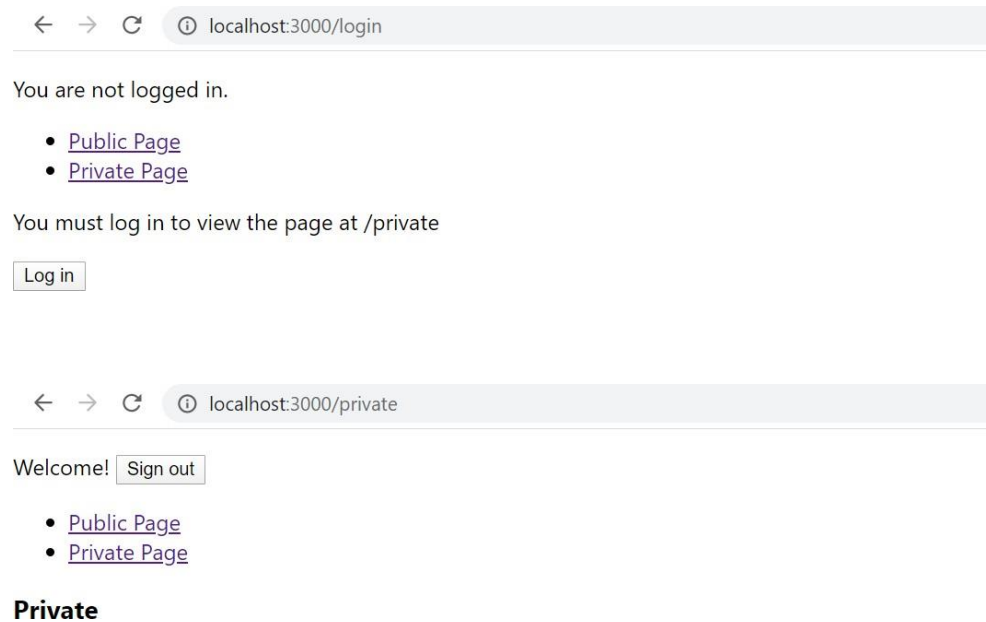
```

```

115
116 function LoginPage() {
117   let history = useHistory();
118   let location = useLocation();
119
120   let { from } = location.state || { from: { pathname: "/" } };
121   let login = () => {
122     fakeAuth.authenticate(() => {
123       history.replace(from);
124     });
125   };
126
127   return (
128     <div>
129       <p>You must log in to view the page at {from.pathname}</p>
130       <button onClick={login}>Log in</button>
131     </div>
132   );
133 }

```

3. Cek hasilnya, ketika klik private page maka diharuskan untuk login terlebih dahulu, ketika sudah masuk dan klik private page kembali kita tidak perlu login kembali karena kita berada dalam posisi logged.



1.2 Soal Latihan Praktikum

1. Buatlah sebuah situs marketplace(online shop) sederhana, dengan menerapkan konsep SPA (Single Page Application) React Router dimana beberapa menu nya jika di klik terdapat fitur "Nesting" dan "Redirect (Auth)".

*****Selamat Mengerjakan*****